SPECIFICATION

TITLE

"METHOD, COMPUTER AND COMPUTER PROGRAM MODULES TO TRANSFER DATA IN A COMPUTER NETWORK, AND SUCH A COMPUTER NETWORK"

BACKGROUND

The invention concerns a method, a computer and computer program modules with which data can be transferred between various components of a computer network. In particular, the invention concerns a method, a device system and computer program modules with which print data are be transferred between a plurality of devices, stored and retrieved.

In high-capacity printing systems, in variable print data and resources (forms, fonts etc.) are combined into what are known as print jobs and are transferred from a data source in a print production system. Such systems (which comprise computers, control units and print devices) are, for example, specified in the publication "Das Druckerbuch", published by Dr. Gerd Goldmann (Océ Printing Systems GmbH), 6th edition (May 2001), ISBN 3-00-001019-x. In chapter 14, the server system Océ PRISMApro is specified. This flexible print data server system is, for example, suited to convert print data from data sources (such as a source computer that provides print data in a specific print data language such as AFP (Advanced Function Presentation), PCL (Printer Command Language), PostScript, SPDS (Siemens Print Data Stream) or LCDS (Line Coded Data Stream)) and to transfer it to a print production system. The print data server system is

thereby comprised of at least one master print server and possibly a few slave print servers. The print data can be read in by the master print server from various sources, if necessary converted, transmitted to the slave print servers, and from there transferred in parallel to a plurality of printers. Depending on the type of data and the involved components, different requirements with regard to the data throughput and the data preservation between the involved components exist for the overall data transmission. This requires a flexible method of data transmission that enables a transmission with high data throughput, if necessary ensuring the preservation of the data for a later access, can be applied across computer boundaries, and is easy to integrate into existing methods.

SUMMARY

It is an object to be able to implement the data transmission between various components in a network of servers, optimized for time and controlled by applications.

In a method and system to transfer data in a network of servers, a computer program module supplying data from a first server is provided. A reading computer program module is provided that reads the supplied data. One of the following transmission modes is selected:

- a complete storage of the data in a file occurs before the reading program module reads the data,
- a segment-by-segment storage of the data in a file occurs such that the reading computer program module already begins with a reading of a

segment while the supplying computer program module is still supplying data, and

- a direct transmission of the data between the supplying computer program module and the reading computer module occurs without buffering.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 illustrates a flow diagram for processing of print data;

Figure 2 illustrates a print data server network; and

Figure 3 illustrates the print data server network of Figure 2 in another operating mode.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

For the purposes of promoting an understanding of the principles of the invention, reference will now be made to the preferred embodiments illustrated in the drawings and specific language will be used to describe the same. It will nevertheless be understood that no limitation of the scope of the invention is thereby intended, such alterations and further modifications in the illustrated device, and such further applications of the principles of the invention as illustrated therein being contemplated as would normally occur to one skilled in the art to which the invention relates.

A method to transfer data in a network of servers is inventively provided, whereby a computer program module supplying the data from a server is provided, and whereby a reading computer program module is provided that reads the supplied data, whereby one of the following transfer modes is selected:

(a) a complete storage of the data in a file occurs before the reading computer program module reads the data,

(b) a segment-by-segment storage of the data in a file occurs such that the reading computer program module already begins with the reading of a segment while the supplying computer program module is still supplying data, and

(c) a direct transmission of the data between the supplying computer program module and the reading computer program module is provided without buffering.

The transmission mode can thereby in particular be selectively controlled by parameters. The reading computer program module and the computer program module supplying the data thereby in particular cooperate via the control parameters.

The computer program module supplying the data is subsequently also called the first component, and the computer program module reading the supplied data is also called the second component. Furthermore, in place of the term server the term computer is also used, and vice versa.

With the preferred embodiment, data can be transferred from a first component of a network of print data servers to a second component in different ways and controlled by the first component. The data are thereby "written" by the first component, as in a storage in a file, and "read" by the second component. The dependence on the ordinary storage of data enables a simple implementation of the functionality by the user. The following

possibilities for data transmission thereby exist, independent of whether the involved components operate on the same or different computers:

1. The transmission actually occurs via a file with a buffering. The first component first writes the data in the storage. After the end of this event, a second component can read this data from the storage. This transmission manner is, for example, advantageous when the second component from the beginning requires the complete data, including the last-written data, for its next work step, as this can be required in specific conversion procedures.

2. The transmission occurs with buffering and what is known as "X-while-X" functionality, meaning that the reading component and processes connected with it (such as, for example, a print event) can already be started while the component supplying the data is still supplying data. The second component thereby begins reading the data, in particular immediately after the first component has begun the writing. Given reading of the data by the second component, it can thereby occur that momentarily no data to be read is present, although the entirety of the data has not yet been transmitted. In an advantageous embodiment, the possibility now exists that the read event of the second component reacts differently to this situation. On the one hand, the possibility exists that the read event is continuously repeated until data can again be read or until it is certain that the first component no longer supplies further data. The second component is then found in a wait state until new data can be read. On the other hand, the read event can in this

case by interrupted and continued at a later point in time. In this type of data transmission, the data remain in the storage after the transmission in order to, for example, by re-read at a later point in time.

3. The transmission occurs directly between components without buffering. The possibility thereby likewise exists for the second component to always repeat the read event in the event that data can be no longer be read before the end of the data transmission is achieved, or to abort the read event and subsequently to restart it.

The necessary manner of the data transmission can be controlled as needed by each component. If the overall event is the execution of a print job, the manner of the transmission can be established for each involved component via the corresponding control data (job ticket data).

Furthermore, the possibility exists to transmit the data in an established block format. The reading component in this case receives the data in the form of specific data blocks that have been determined by the writing component. In this manner it is guaranteed that the reading component receives the data in precisely the block format as the writing component provides it. A subsequent merging or division of the data in the reading component is thus prevented.

If the data are stored in the transmission, further information about this file can be queried. The temporary read or write position in the file can be queried. The size of the file can be determined. The possibility exists to test whether the file end has been achieved with current position. Differentiation is thereby made between the actual file end after the end of the writing of the

data by the writing component and a preliminary file end that was already reached by the reading component given the "X-while-X" functionality, but in expectation of further data from the writing component does not yet represent the actual file end.

In the case of the storage of the data, this functionality can be newly positioned within the data. This functionality is, for example, necessary when the data processing must be restarted to continue at a specific point of the data stream after the interruption of a print job.

In the preferred embodiments, respectively corresponding functions are executed by both components in addition to a data transmission that uses the manner of transmission with "X-while-X" functionality shown above. This is not possible for the writing component because the data are transmitted, for example, from an outside system, thus a further component can mark the file created by the writing component as "writing" without, however, writing data in the file itself. The reading component can now receive the data with the "X-while-X" functionality.

All previously described functionalities of the preferred embodiments can be applied in the framework of the data transmission between two components on the same print server or between two components on two different print servers networked with one another. The writing component can thereby write the data only on its print server, but the reading component can receive the data from it or from a print server networked with its print server.

A further aspect of the preferred embodiments concerns the situation that data is no longer present for the reading program. Differentiation is thereby made between both of the cases 2 (transmission with buffering) and 3 (direct transmission without buffering) cited above. In case 2, the user in any case receives a corresponding message that data is no longer available. The user can then decide whether the read is to be continually retried in a loop or to abort the read attempt and, if necessary, to implement other work processes and continue the read at a later point in time. In the case 3, a parameter "non-block" can be specified upon opening the file. Depending on whether this parameter is assigned, given the direct transmission either a notice is delivered to the user that at the time it cannot be read, or (when the parameter is not set) the reading process waits a default time until data is again present for reading.

The difference between the procedures of both cases 2 and 3 is thus that in the second case, the user must decide, while in the third case, when the parameter is not set, the read process automatically goes into the wait state. The functionality supported in the third case can in particular be supported in that the first component and the second component are connected with one another via a operating system-specific or operating system-proprietary functionality, what is known as a socket connection, which for its part offers such a possibility.

Furthermore, a computer program module is provided that effects a inventive method procedure upon loading and execution on a computer in cooperation with a further computer program module. Both computer

computer or essentially run on two different computers that are connected with one another via a network connection.

Furthermore, a computer a system to implement the method is provided, as well as a computer network with a system to implement the method.

It is illustrated in Figure 1 how print data are processed in a print data server. The print data are thereby read in via an input interface 2, and in particular are filtered via an input filter 3 that checks the data in terms of its data integrity and, as necessary, undertakes changes. In an independent process, control data with which the processing or further processing of the print data can be controlled are generated and/or provided in a print job manager 4. The control data directly act on the processes in a job distribution system (order distribution system) 5. Data that have been filtered in the input filter 3 are then written in a print filter 6. From there, the print data are transferred to an output interface 7 in a read process (FTR-read). From there, they are either supplied to another print data server for further processing or are directly output to a printer for printing.

The write process (FTR-write) in the print file 6 that is executed by a writing computer program module and the read process (FTR-read) from the print file 6 that is executed by a reading computer program module are thereby designed such that the transmission of the data from the input filter 3 to the output interface 7 occurs controlled by a parameter that has been predetermined in the print job manager 4, with the following possibilities:

1) Either the transmission occurs via the print file 6 with a buffering, whereby first all print data of a print job are stored in the buffer before the print data are read out into the output interface 7. This transmission method is, for example, necessary when a subsequent process that taps the data at the output interface 7 requires the complete data of the print job at the start.

2) As an alternative to this, the transmission of the data from the input filter 3 to the output interface 7 can occur such that, during the writing of data from the input filter 3 into the print file 6, other data that have already been stored beforehand are read out from the print file 6 into the output interface 7. When, for example, the print data are directly transferred to a print device from the output interface 7, what is known as a Print-While-Spool functionality can thereby be realized in which the first print data are already printed out while subsequent print data are still being written in the print file 6.

3) Finally, the possibility exists that the print data are directly transferred from the input filter 3 to the output interface 7, whereby both computer program modules that control the write event (FTR-write) and the read event (FTR-read) are designed such that the direct transmission is possible without buffering. A storage in the print file 6 thereby does not occur.

In Figure 2, a network 12 is shown in which a first print data server 10 is connected with a second print data server 11 via a network connection 18. Print data are read into the first print data server 10 via an input interface 13. To transmit the print data from the first print data server 10 to the second print data server 11, a first computer program module that controls a write process 15 in the first print data server 10 cooperates with a reading computer

program module that controls a read process 16 in the second print data server 11. This effects a process connection 14 of the read process 16 with the write process 15. A control parameter determines the type of data transmission between the two print data servers 10, 11. The write process 15 which receives the data via the interface 13, for example via a download process, provides these data to the read process 16. There are thereby three variations as described hereafter.

In a first variation, the data are completely written as a file on the fixed disc 19 of the first print data server 10 and the read process subsequently reads the data from the fixed disc 19. The write process 15 thereby operates locally on the first print data server 10, while the read process 16 can read across computers, thus itself runs on the second print data server 11 but reads the data from the first print data server 10 from its fixed disc 19. The read process 16 can then output these data, for example to a print device 17 as shown in Figure 2, or can transfer the data to another system or store the data on a fixed disc in the second print data server 11.

In a second variation, the read process 16 already reads data from the fixed disc 18 of the first print data server 10 via the network connection 18 while the write process 15 is still writing data to the fixed disc 19 (x-while-x process).

In order to transfer data across computers from the print data server 10 to the print data server 11, a read process dependent on the read process 16 of the second print data server is set up on the first print data server, the dependent read process being connected with the read process 16 of the

second print data server via a socket connection. The read process in the first print data server receives from the read process 16 of the second print data server a trigger signal to read the data, and sends this back to the read process 16 via the socket connection.

With regard to control characters, a file-end control character (End of File, EoF) and a data-end control character (End of Data, EoD) are provided. Given the presence of the EoF control character, it is indicated that the writing process 15 is concluded and the file is closed. The reading process is then likewise closed. Given the presence of the EoD control character, it is indicated that the writing process 15 is writing further data. The file remains open and the reading process waits for subsequent data and leaves the file open.

When the socket connection cited above exists, the buffer file is opened for reading a parameter that establishes the various further handling possibilities when data is no longer present. On the one hand, a waiting action can thereby be triggered, or on the other hand a request can be issued to the user in order to initiate selectable further process steps.

The write process 15 outputs an EoD control signal when a maximum file size (buffer size) is reached.

In Figure 3, the arrangement of Figure 2 is shown, whereby here a socket connection exists between the computer program modules that implement the write process 15 or the read process 16. Based on this connection, the print data are forwarded from the interface 13 directly to the print device 17 without buffering on the fixed disk 19. The data are

transferred without buffering between the writing and reading computer program modules. The remaining options and functions of this operating mode already specified above can also hereby be used. The writing computer program module and the reading computer program module are thereby directly connected via a computer-internal fixed interface (PORT) that can also be used across computers.

Although the system and method were predominantly specified in examples in a print production environment, it can also be used just as well for other data-technical systems in which data must be exchanged in a computer network.

The system and method are in particular suited to be realized as a computer program (software). In particular, it comprises the supplying computer program module and the reading computer program module, and can be distributed as a file or file collection on a data medium such as a diskette or CD-ROM, or via a data or, respectively, communication network. Such and comparable computer program products or computer program elements are embodiments of the system.

The system and method can be applied in a computer, in a print data or in a print system with upstream or downstream data processing devices. It is thus clear that corresponding computers on which the system and method are used can comprise further known technical devices such as input devices (keyboard, mouse, touchscreen), a microprocessor, a data or control bus, a display device (monitor, display) as well as a working storage, a fixed disc storage and a network card.

While the invention has been illustrated and described in detail in the drawings and foregoing description, the same is to be considered as illustrative and not restrictive in character, it being understood that only the preferred embodiments have been shown and described and that all changes and modifications that come within the spirit of the invention are desired to be protected.